Lab 02: To perform Data Cryptography and De-cryptography

Note: ***Students must demonstrate their lab in the following week of October 5th, 2020 to get the grade. This lab is worth 20marks. A Missed demonstration will result in 25% grade deduction. Late submission and late demonstration to the lab is welcomed but with a loss of marks. That is, you do not earn full marks for late submission. 10% penalty for every day late up to 50%. Work will not be accepted after 10 days.***
You must talk to your lab professor in case of any unprecedented situation which may result in late submission.
You are required to submit the screenshot of your terminal showing successful execution of the program (including relevant display messages or output statements) along with properly commented (as per assignment submission standards doc on BrightSpace) text file of your code.

Setup:

1. Create a directory **`Lastname02`**. You are going to develop your lab here.

Network Cryptography

The unpredictable growth of Internet communications and data storage on Internet-connected computers has greatly increased privacy concerns. The field of cryptography is concerned with coding data to make it difficult (and hopefully—with the most advanced schemes—impossible) for unauthorized users to read. In this exercise you will investigate a simple scheme for encrypting and decrypting data.

Problem Statement: A company that wants to send data over the Internet has asked you to write a program that will encrypt it so that it may be transmitted more securely. All the data is transmitted as four-digit integers. Your application should read a four-digit integer entered by the user and encrypt it as follows:

- Replace each digit with the result of adding 7 to the digit and getting the remainder after dividing the new value by 10.
- Then swap the first digit with the third and swap the second digit with the fourth.
- Then print the encrypted integer.
- Write a separate application that inputs an encrypted four-digit integer and decrypts it (by reversing the encryption scheme) to form the original number.

Program #1:

Write a small C program **`crypto.c`** that:

1. Ask the user an integer number
2. Reads an integer number
3. Checks that the number is at least **`4`** digits long
4. If the number of more than **`4`** digits, it should print a message to the end user "**`Invalid number of digits`**" and terminate with an exit value of **`EXIT_FAILURE`**
5. If the number of digits is correct, your program should encrypt the number with the encryption algorithm given above.
6. At the end, your program should print the encrypted number and terminate with a value of **`EXIT_SUCCESS`**.

The following demonstrates the execution of the program:

```
desktop@bahris:CST8234/# ./crypto
Enter a 4 digit integer number:  1256
Encrypted number of 1256 is 2389
desktop@bahris:CST8234/# echo $?
0
desktop@bahris:CST8234/# ./crypto
Enter a 4 digit integer number:  123456
Invalid number of digits 6 in 123456
desktop@bahris:CST8234/# echo $?
1

                          SAMPLE TEST OUTPUT:  crypto
```

To successfully complete this program and obtain all the marks, you will need to:

1. Use the macros **`EXIT_FAILURE`** and **`EXIT_SUCCESS`** define in the library **`stdlib.h`** to indicate unsuccessful or successful termination of your program
2. Define **`DIGITS`** as a macros in your program. **`DIGITS`** should be defined as **`4`**.
3. Write a function, with function prototype **`int crypto( int )`**
   - **`crypto( )`** should encrypt a 4 digit number as establish above.
   - Your function should return the encrypted number.
     - **HINT**: Separate the digits and store them in an array.
4. Your **`main()`** should:
   - Use the function **`scanf( )`** to read an integer from the keyboard.
   - Validate the number entered by the user.
   - If the number has more than 4 digits, your program should print an error message and terminate with **`EXIT_FAILURE`**
   - If the number has less than 4 digits, your program should invoke a function to encrypt the number.
   - Print the newly encrypted number and exit with **`EXIT_SUCCESS`**
5. Your program should be compiled with the flags **`-Wall -ansi -pedantic`**

Program #2:

Copy your **`crypto.c`** to a new program **`decrypt.c`**. Your decrypt program should be able to decrypt an encrypted number.

```
desktop@bahris:CST8234/# ./decrypt
Enter a 4 digit integer number:  2398
Decrypted number of 2398 is 2156

                         SAMPLE TEST OUTPUT:  decrypt
```